**Udacity Project readme file**

This repo contains the starter materials for projects from the Udacity Azure Cloud DevOps Nanodegree Program.

**This document is to setup environment for deploying web server on Azure:**

**Overview of this project:**

Here are the steps of instructions to create infrastructure code of terraform template, packer template and deploy them to website using load balancers.   
  
All these steps can be achieved either by azure portal directly or using AzureCLI or PowerShell commands.   
To reduce the redundancy and to follow DRY concept, it's always better practice to do it with AzureCLI.

**Environment setup:**

1) We should create an Azure account if we don't have one.

2) Install terraform and write a template to build infrastructure of code.

3) Install packer and create a packer template to deploy Vm's

4)Install Azure CLI

**Deploying Policy:**  
Azure policy evaluates resources in Azure by comparing the properties of those resources to business rules. We have to create our own azure policy and we can set our own rules which suits our buisness.  
Here I have created a policy definition to deny the creation of resources that do not have tags. And I have named my policy as “tagging-policy”.

We have set of rules to create this policy.[Creating Azure policy and managing it.](https://docs.microsoft.com/en-us/azure/governance/policy/tutorials/create-and-manage)   
  
**Packer Template:**

Packer template helps us to create image of server, and which can be reused to deploy many resources.   
To create my packer template I followed below steps:

1) Initially I created a resource group in Azure portal.

2) Create a service principal using AzureCLI

3) Define packer template

4) Created a template for custom VM in json file:

5) Build the packer image using ./packer build demo.json

Output on portal  
   
image with linux\_june is created in my Udacityproject1 resource.

5) **Terraform Template:**

Terraform template can also be creating directly on portal, but I choose to create one by using script.  
And this template contains two files namely main.tf and variables.tf.  
In main.tf we can specify what type of resources are we interested to create using this template and variables.tf can help us to reuse this template by creating variables instead of hardcoding the values.. And below is the screenshot of template which I used to deploy for my project “Udacityproject1”.

Steps to create this template and deploy it:

A) Create template

B) run the following commands: terraform init (to initialize terraform)

C) terraform plan (to build the plan)

D) enter the values required for project which are prompted with variables.tf file.

E) once the plan is executed, it will prompt us to create the plan or edit it.

F) Terraform apply (to deploy the plan)

G) terraform plan –out (which helps us to print the plan)

H) To make it DRY, add script with count variable, which helps you to create a greater number of virtual machines at same time. And make appropriate changes in variables.tf  
This is one advantage creating through terraform template than creating VM’s using portal.

New VM, Virtual network, Network interface and Disk are created with terraform plan “Udacitydemo”

6) Creating load balancer:

Load balancers help to balance network between inbound and outbound resources.  
I created one load balancer for my project and assigned few resources VM’s as outbound connections.

7)And finally we have to destroy the terraform created using Terraform destroy

it will prompt to get it confirmed :

And please confirm the number of options in destroy option and confirm yes